Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Marko Ambrožič

# Dinamična izbira metod za profiliranje spletnih uporabnikov

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Zoran Bosnić

Ljubljana, 2014

*Besedilo je oblikovano z urejevalnikom besedil LaTeX.*

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Profiliranje spletnih uporabnikov postaja pomembno področje razvoja spletnih aplikacij, saj omogoča sledenje uporabnikov, učenje njihovih interesov in s tem zagotavljanje personalizirane uporabniške izkušnje. Aktualno raziskovalno delo predstavlja več različnih metod, ki pa imajo različne uspešnosti izdelave profila glede na čas opazovanja uporabnika in obseg profila.

V diplomskem delu naj kandidat preuči načine kombiniranja različnih metod za profiliranje uporabnikov, pri čemer naj izhaja iz metodologje avtorjev Košir in sod. (2013). V diplomi naj predlaga metodo za dinamično izbiro metode profiliranja, s katero naj preseže uspešnost samostojnih metod. Rezultate naj prikaže in ovrednoti.

# Izjava o avtorstvu diplomskega dela

Spodaj podpisani Marko Ambrožič, z vpisno številko **63090095**, sem avtor diplomskega dela z naslovom:

*Dinamična izbira metod za profiliranje spletnih uporabnikov*

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom izr. prof. dr. Zorana Bosnića,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 25. avgusta 2014 Podpis avtorja:

# Kazalo

# Seznam uporabljenih kratic

| **kratica** | **angleško** | **slovensko** |
|---|---|---|
| **CA** | classification accuracy | klasifikacijska točnost |
| **SVM** | support vector machine | metoda podpornih vektorjev |
| **adv** | advertising agency profiling domain | domena profilov oglaševalske mreže |
| **mdl** | e-learning environment profiling domain | domena profilov uporabnikov spletne učilnice |

# Povzetek

Profiliranje uporabnikov postaja vse bolj pomembna tema pri razvoju spletnih strani, saj omogoča zagotavljanje boljše uporabniške izkušnje z ugotavljanjem uporabnikovih interesov. V tem delu se ukvarjamo z dinamično izbiro metod profiliranja uporabnikov. Cilj je uporabiti metode strojnega učenja in zgraditi učni model, ki bo znal kar najbolje kombinirati metode profiliranja in tako ustvariti kombinirano metodo profiliranja, uspešnejšo od vsake posamezne metode, ki smo jih uporabili pri učenju. Pokazali smo, da je kombiniranje profilirnih algoritmov z uporabo strojnega učenja lahko močno orodje pri izboljšavi uspešnosti profiliranja. Pokazali smo tudi, da je uporaba dinamične izbire metod smiselna v primeru, ko so razlike med posameznimi algoritmi profiliranja večje in so tako tudi možnosti za izboljšave večje.

**Ključne besede:** dinamična izbira metod, profiliranje uporabnikov, strojno učenje.

# Abstract

User profiling is becoming an increasingly important subject in the field of web development as it enables improving the user experience through learning the users interests. In this study we examine dynamic selection of web user profiling methods. Our goal is to use machine learning methods to build a learning model that predicts the most successful combined profiling method, which is expected to be significantly better from each individual method. We have shown that combining of profiling methods using machine learning can be a powerful tool when looking for a way of improving the accuracy of web user profiles. We have also shown that dynamic selection is most effective when differences between profiling methods are relatively large and therefore providing room for improvement.

**Keywords:** dynamic method selection, user profiling, machine learning.

# Poglavje 1

# Uvod

Profiliranje uporabnikov je področje informatike, ki je v zadnjih letih doživelo velik razmah. Predvsem neinvazivne metode, ki omogočajo profiliranje z analizo velike količine podatkov, zbranih brez interakcije uporabnika. Ti podatki so lahko ime, starost, pretekla dejanja, itd. Motivacija za gradnjo čim bolj natančnih uporabniških profilov je spoznanje, da so preference, cilji in želje uporabnikov različne. Z odkrivanjem teh razlik lahko zagotovimo boljšo uporabniško izkušnjo, prilagojeno uporabniku.

V [6] je predstavljenih več različnih metod za profiliranje uporabnikov. Za gradnjo profilov uporabljajo algoritem AverageAction (opisano v poglavju 2), ki ga dopolnjuje časovno pozabljanje. S spreminjanjem stopnje pozabljanja lahko zgradimo več različnih metod. Te metode so različno uspešne pod različnimi pogoji. Z dinamično izbiro metod bomo poskusili doseči, da se pod določenimi pogoji vedno izbere najbolj optimalna. Dinamično izbiro bomo dosegli z uporabo klasifikacije.

Klasifikacija za učenje uporablja klasifikacijska pravila. Ta so sestavljena iz pogojnega in sklepnega dela. Pogojni del je konjunkcija pogojev $C_i$, sklepni del pa vsebuje pogoj $C_0$.

$$C_{i_1} \ \& \ C_{i_2} \ \& \ ... \ \& \ C_{i_l} \implies C_0 \tag{1.1}$$

Pogoji so oblike $C_i = (A_i \in V_i^{'})$, kjer je $V_i^{'}$ podmnožica možnih vrednosti atributa $A_i : V_i^{'} \subset V_i$.

Najbolj uspešne izmed klasifikacijskih algoritmov bomo uporabili za učenje iz podatkov povprečja uspešnosti profiliranja. Za najbolj uspešne so se v našem primeru izkazali naivni Bayes, C4.5, naključni gozd, logistična regresija, algoritem bagging in glasovanje s povprečenjem verjetnosti modelov J48, naivni Bayes in naključni gozd. Podatki so nam na voljo v štirih matrikah velikosti $11 * 11$ in združeni po starosti profila ter velikosti hranjene zgodovine profilov uporabnikov. Pridobljeni so bili s profiliranjem uporabnikov oglaševalske mreže. To domeno podatkov bomo v nadaljevanju označevali z oznako ADV.

Rezultate bomo primerjali na podlagi klasifikacijske točnosti in povprečja uspešnosti profiliranja pri vsaki kombinaciji parametrov starost profila / zgodovina ter skupni povprečni vrednosti uspešnosti profiliranja. Poleg tega bomo izbrane klasifikacijske algoritme preizkusili tudi na drugi domeni podatkov, pridobljenih s profiliranjem študentov, uporabnikov spletne učilnice. To domeno podatkov bomo v nadaljevanju označevali z ozanko MDL.

Dosežene izboljšave bomo na koncu potrdili tudi z Wilcoxonovim statističnim testom in s tem preverili, če so naše izboljšave statistično značilne.

V 2. poglavju bomo najprej predstavili profilirne metode, ki jih želimo kombinirati z uporabo klasifikacije. V poglavju 3 bomo predstavili podatke, ki so nam na voljo, izpostavili nekaj značilnosti teh podatkov ter opisali način, s katerim smo te podatke preoblikovali za namene klasifikacije. V 4. poglavju bomo opisali uporabljene metode strojnega učenja in mere, ki smo jih uporabili za ocenjevanje uspešnosti klasifikacije. V poglavju 5 bomo predstavili rezultate uspešnosti klasifikacije, primerjavo učenja na obeh domenah podatkov in primerjali doseženo izboljšano dinamično metodo z vsako izmed

statičnih metod, katerih podatki so nam bili na voljo na začetku. V poglavju 5 so predstavljeni tudi podatki statističnega testa.

# Poglavje 2

# Opis metod za profiliranje uporabnikov

V tem poglavju bomo predstavili profilirne metode, s katerimi so bili pridobljeni podatki, ki smo jih uporabili pri strojnem učenju za dosego dinamične izbire metod. Opisane metode so bile prvič predstavljene v [6].

Vse uporabljene metode izhajajo iz osnovne metode AverageAction. Ta gradi ontološki profil uporabnika na podlagi njegovih preteklih akcij. Uporabnikove akcije so definirane kot obiskane spletne strani, ki so predhodno kategorizirane v drevesno strukturo (ontologijo), v kateri vsako vozlišče predstavlja tematsko kategorijo, ki je posplošitev tematskih kategorij v poddrevesu. Za vsako akcijo uporabnika algoritem poveča oceno ustrezni kategoriji. Vse ocene so shranjene v vektorju velikosti števila vseh tematskih kategorij. Na koncu metoda ocene normalizira, tako da je njihova vsota enaka 1.

Vzemimo za primer spletno mesto, ki vsebuje tri spletne strani. Obisk vsake izmed teh strani predstavlja akcijo, ki jo lahko uporabnik izvede. Predpostavimo, da so vse akcije predhodno kategorizirane v ontologijo tematskih kategorij in da vsaka akcija spada v svojo ločeno kategorijo. Predpostavimo tudi, da hranimo zadnjih deset akcij uporabnika, ki so shranjene v normaliziranem vektorju, prikazanem v enačbi 2.1.

$$norm([5, 4, 1]) = [0.5, 0.4, 0.1] \tag{2.1}$$

Ker je bilo največ obiskov prve strani, bo njej pripadajoča kategorija podana kot prvi predlog.

Osnovno metodo izboljšuje funkcija časovnega pozabljanja. Funkcija časovnega pozabljanja temelji na spoznanju, da so uporabnikove nedavne akcije bolj pomembne pri ugotavljanju trenutnih interesov kot tiste, ki so se zgodile v bolj oddaljeni zgodovini. S tem namenom je bila vpeljana logaritmična funkcija časovnega pozabljanja, ki dodeljuje pomembnost posameznim akcijam. Zadnja uporabnikova akcija je najpomembnejša in ji je zato dodeljena vrednost $importance_{\text{action}} = 1$. Vrednost pomembnosti starejših akcij je izračunana po formuli (2.2). S takšnim dodeljevanjem večje pomembnosti nedavnim akcijam je doseženo boljše prilagajanje spreminjanju uporabnikovih interesov.

$$importance_{\text{action}} = \frac{1}{1 + log_a(age_{\text{action}} + 1)} \tag{2.2}$$

S spreminjanjem baze logaritma a, lahko nadzirmo hitrost pozabljanja. Manjše vrednosti pomenijo hitrejše pozabljanje. Z $age_{\text{action}}$ označimo starost akcije, ki pomeni pretečen čas od njene pojavitve.

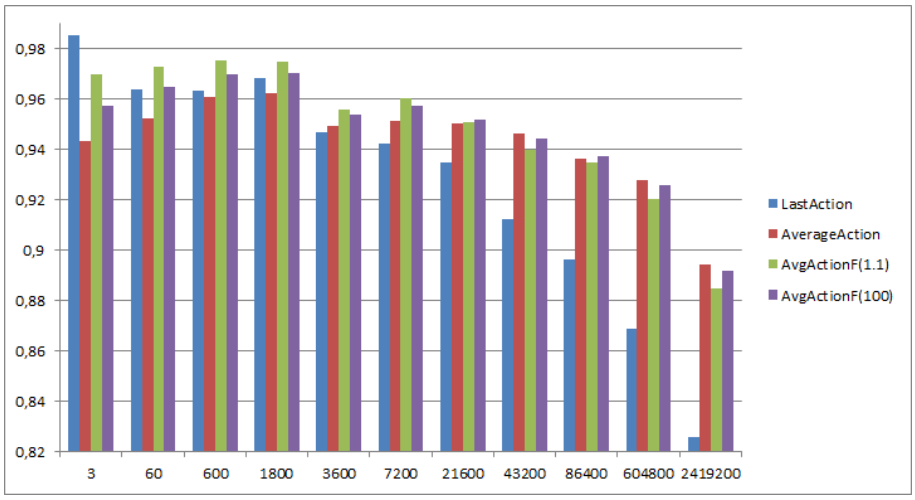
Z uporabo funkcije časovnega pozabljanja lahko dosežemo naslednje tri načine delovanja profilirnega algoritma:

1. Profiliranje brez pozabljanja, imenovano **AverageAction**. Tu funkcija časovnega pozabljanja nima vpliva.

2. Profiliranje z zmernim pozabljanjem, imenovano **AverageActionF(a)**. S spreminjanjem parametra $a$ lahko nadziramo moč pozabljanja. V tem delu uporabljamo podatke za dve različici tega načina delovanja. **AverageActionF(1.1)** in **AverageActionF(100)**, ki sta predstavljeni v [6]

3. Profiliranje s popolnim pozabljanjem, imenovano **LastAction**. Ta način vedno hrani samo zadnjo izvršeno akcijo.

Kot bo vidno tudi v nadaljevanju je profiliranje brez pozabljanja najbolj uspešno pri napovedovanju uporabnikovih dolgoročnih interesov, profiliranje z zmernim pozabljanjem pri napovedovanju srednjeročnih interesov in profiliranje s popolnim pozabljanjem pri napovedovanju kratkoročnih interesov. Slika 2.1.




Slika 2.1: Primerjave uspešnosti štirih načinov delovanja profilirnega algoritma AverageAction za dolžino hranjene zgodovine 100, glede na starost profila v sekundah. Metoda LastAction je uspešna le v prvih nekaj sekundah, nato jo zamenja metoda AverageActionF(1.1). Pri daljših obiskih je najprimernejša metoda AverageAction.

# Poglavje 3

# Opis podatkovnih množic

Za strojno učenje so nam bile na voljo podatkovne množice v obliki matrik velikosti 11x11 za vsako izmed izbranih profilirnih metod. V stolpcih so vrednosti razvrščene po starosti profila, ki je pretečeni čas od izgradnje profila do ocene njegove uspešnosti, v vrsticah pa vrednosti, razvrščene po dolžini hranjene zgodovine profila, ki pomeni število uporabljenih akcij pri gradnji profila.

| | 5 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0,951955 | 0,941526 | 0,93998 | 0,935983 | 0,930143 | 0,937889 | 0,944987 | 0,936627 | 0,938565 | 0,929098 | 0,943249 |
| 60 | 0,948436 | 0,950895 | 0,955453 | 0,955704 | 0,952721 | 0,949624 | 0,952463 | 0,948082 | 0,947076 | 0,959453 | 0,95221 |
| 600 | 0,946317 | 0,952353 | 0,957362 | 0,95292 | 0,953156 | 0,955431 | 0,961689 | 0,961174 | 0,964671 | 0,960534 | 0,960611 |
| 1800 | 0,931518 | 0,940452 | 0,944592 | 0,943681 | 0,950357 | 0,953462 | 0,960736 | 0,961393 | 0,96579 | 0,962455 | 0,962286 |
| 3600 | 0,925337 | 0,937548 | 0,944252 | 0,91203 | 0,920214 | 0,940071 | 0,943449 | 0,955336 | 0,963492 | 0,959885 | 0,949157 |
| 7200 | 0,912027 | 0,916238 | 0,93639 | 0,943429 | 0,93654 | 0,950314 | 0,94562 | 0,960026 | 0,960213 | 0,950664 | 0,951347 |
| 21600 | **0,919811** | **0,926933** | **0,926937** | **0,922651** | **0,935868** | **0,943715** | **0,931321** | **0,929726** | **0,925274** | 0,945969 | 0,950458 |
| 43200 | **0,892587** | **0,903944** | **0,91764** | **0,92767** | 0,932745 | 0,932707 | 0,93528 | **0,94597** | **0,950174** | **0,953682** | **0,946167** |
| 86400 | **0,888905** | **0,914164** | **0,922776** | **0,920868** | **0,940592** | **0,935704** | 0,941768 | 0,94299 | 0,945855 | 0,930153 | 0,936443 |
| 604800 | **0,877567** | **0,890932** | **0,906609** | **0,915066** | **0,916506** | **0,919781** | **0,918595** | **0,924017** | **0,925678** | **0,926802** | **0,927984** |
| 2419200 | **0,837387** | **0,855408** | **0,870129** | **0,877645** | **0,88232** | **0,886985** | **0,889771** | **0,890763** | **0,892014** | **0,896272** | **0,894444** |

Tabela 3.1: Podatkovna množica za metodo AverageAction. V vrsticah so podatki glede na starost profila v sekundah, v stolpcih pa glede na dolžino hranjene zgodovine oz. število akcij. V primerjavi z ostalimi podatkovnimi množicami je razvidno, da je ta metoda uspešnejša pri daljših obiskih, torej napovedovanju dolgoročnih interesov. Odebeljene so uspešnosti, ki so najvišje glede na istoležne celice v tabelah 3.1 - 3.4.

Z namenom uporabe pri klasifikaciji smo podatke preoblikovali v obliko (zgodovina, starost profila, razred), kjer je zgodovina število hranjenih obi-

| | 5 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0,97685 | 0,96800 | 0,96619 | 0,95949 | 0,95605 | 0,96121 | 0,96344 | 0,95564 | 0,95730 | 0,94543 | 0,95739 |
| 60 | 0,95725 | 0,96316 | 0,96623 | 0,96873 | 0,96682 | 0,96369 | 0,96593 | 0,96439 | 0,96020 | 0,96960 | 0,96490 |
| 600 | **0,95536** | 0,96441 | 0,96905 | 0,96330 | 0,96651 | 0,96786 | 0,97233 | 0,97139 | 0,97234 | 0,96847 | 0,96994 |
| 1800 | **0,93851** | **0,94932** | 0,95610 | 0,95732 | 0,96239 | 0,96385 | 0,96976 | 0,96965 | 0,97368 | 0,97223 | 0,97044 |
| 3600 | **0,93157** | **0,94240** | **0,94711** | 0,92589 | 0,93195 | 0,94928 | 0,94883 | 0,96221 | 0,97026 | 0,96853 | 0,95407 |
| 7200 | **0,90836** | **0,91771** | **0,93983** | **0,94283** | **0,94017** | **0,95050** | 0,95209 | 0,96487 | 0,96624 | 0,95564 | 0,95712 |
| 21600 | 0,91704 | 0,92529 | 0,92216 | 0,91596 | 0,93530 | 0,94361 | 0,92939 | 0,92596 | 0,92106 | **0,94733** | **0,95199** |
| 43200 | 0,89182 | 0,90113 | 0,91622 | 0,92562 | **0,93497** | **0,93323** | **0,93950** | 0,94418 | 0,94875 | 0,95348 | 0,94423 |
| 86400 | 0,88352 | 0,91332 | 0,91919 | 0,91521 | 0,94072 | 0,93432 | **0,94294** | **0,94304** | **0,94633** | **0,93079** | **0,93715** |
| 604800 | 0,86930 | 0,88846 | 0,90343 | 0,91280 | 0,91439 | 0,91867 | 0,91776 | 0,92315 | 0,92275 | 0,92448 | 0,92583 |
| 2419200 | 0,82960 | 0,84935 | 0,86407 | 0,87171 | 0,87846 | 0,88264 | 0,88610 | 0,88922 | 0,88862 | 0,89528 | 0,89186 |

Tabela 3.2: Podatkovna množica za metodo AverageActionF(100). V primerjavi z ostalimi podatkovnimi množicami vidimo, da je ta metoda boljša pri napovedovanju srednjeročnih interesov v kombinaciji z metodo AverageActionF(1.1). Odebeljene so uspešnosti, ki so najvišje glede na istoležne celice v tabelah 3.1 - 3.4.

| | 5 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0,98531 | 0,97858 | 0,98168 | 0,97410 | 0,97654 | 0,98009 | 0,97760 | 0,97125 | 0,97117 | 0,95968 | 0,96976 |
| 60 | **0,95746** | **0,96534** | **0,96943** | **0,97329** | **0,97266** | **0,96972** | **0,97109** | **0,97213** | **0,96573** | **0,97493** | **0,97307** |
| 600 | 0,95501 | **0,96587** | **0,97181** | **0,96573** | **0,97172** | **0,97315** | **0,97553** | **0,97528** | **0,97491** | **0,97326** | **0,97555** |
| 1800 | 0,93805 | 0,94788 | **0,95823** | **0,96163** | **0,96818** | **0,96919** | **0,97237** | **0,97386** | **0,97702** | **0,97770** | **0,97486** |
| 3600 | 0,92900 | 0,93949 | 0,94398 | **0,93177** | **0,93658** | **0,95221** | **0,94934** | **0,96648** | **0,97237** | **0,97175** | **0,95570** |
| 7200 | 0,90043 | 0,91577 | 0,93864 | 0,93696 | 0,93934 | 0,94816 | **0,95484** | **0,96343** | **0,96699** | **0,95742** | **0,96020** |
| 21600 | 0,91242 | 0,92041 | 0,91436 | 0,90683 | 0,93195 | 0,94015 | 0,92425 | 0,91905 | 0,91744 | 0,94571 | 0,95104 |
| 43200 | 0,88927 | 0,89162 | 0,91228 | 0,92171 | 0,93409 | 0,93151 | 0,93669 | 0,94037 | 0,94452 | 0,94933 | 0,93970 |
| 86400 | 0,87465 | 0,90314 | 0,90905 | 0,90557 | 0,93736 | 0,92862 | 0,93954 | 0,93904 | 0,94195 | 0,92909 | 0,93493 |
| 604800 | 0,85766 | 0,87859 | 0,89237 | 0,90251 | 0,90508 | 0,91241 | 0,91118 | 0,91819 | 0,91502 | 0,91864 | 0,92048 |
| 2419200 | 0,81815 | 0,83513 | 0,84829 | 0,85758 | 0,86736 | 0,87143 | 0,87680 | 0,88231 | 0,88036 | 0,88923 | 0,88465 |

Tabela 3.3: Podatkovna množica za metodo AverageActionF(1.1). V primerjavi z ostalimi podatkovnimi množicami vidimo, da je ta metoda boljša pri napovedovanju srednjeročnih interesov v kombinaciji z metodo AverageActionF(100). Odebeljene so uspešnosti, ki so najvišje glede na istoležne celice v tabelah 3.1 - 3.4.

skov v profilu, starost profila pretečen čas od izgradnje profila do ocene uspešnosti in razred koda najuspešnejše metode pri posamezni kombinaciji parametrov zgodovina in čas. Metode AverageAction, AverageActionF(100), AverageActionF(1.1) in LastAction smo označili s kodami a, b, c in d v tem zaporedju. Izsek podatkov je viden v tabeli 3.5.

| | 5 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | **0,990178** | **0,98298** | **0,994047** | **0,983928** | **0,994631** | **0,996437** | **0,99337** | **0,989085** | **0,980114** | **0,973563** | **0,98522** |
| 60 | 0,951659 | 0,957572 | 0,959496 | 0,962838 | 0,966661 | 0,960629 | 0,963323 | 0,960921 | 0,955244 | 0,96902 | 0,963976 |
| 600 | 0,948974 | 0,959152 | 0,962551 | 0,957671 | 0,965436 | 0,963064 | 0,962061 | 0,964765 | 0,966632 | 0,966731 | 0,963254 |
| 1800 | 0,930337 | 0,939158 | 0,950833 | 0,94843 | 0,959692 | 0,959276 | 0,951497 | 0,965904 | 0,971808 | 0,971646 | 0,96847 |
| 3600 | 0,918082 | 0,933427 | 0,923783 | 0,916778 | 0,924257 | 0,942982 | 0,93326 | 0,937997 | 0,966092 | 0,959039 | 0,946565 |
| 7200 | 0,887229 | 0,903821 | 0,917888 | 0,91443 | 0,921889 | 0,929565 | 0,928609 | 0,946913 | 0,949721 | 0,945546 | 0,942102 |
| 21600 | 0,904464 | 0,910239 | 0,888636 | 0,877052 | 0,905821 | 0,914733 | 0,895047 | 0,87484 | 0,891117 | 0,933728 | 0,934584 |
| 43200 | 0,88149 | 0,870104 | 0,887141 | 0,894548 | 0,900196 | 0,900991 | 0,907584 | 0,892239 | 0,918369 | 0,921078 | 0,912149 |
| 86400 | 0,863306 | 0,884681 | 0,880305 | 0,867988 | 0,907341 | 0,898529 | 0,900334 | 0,881872 | 0,900091 | 0,906172 | 0,896251 |
| 604800 | 0,843724 | 0,858504 | 0,856949 | 0,866815 | 0,858169 | 0,870181 | 0,87072 | 0,8728 | 0,85262 | 0,875779 | 0,868898 |
| 2419200 | 0,802359 | 0,811884 | 0,804102 | 0,810214 | 0,821139 | 0,818948 | 0,826633 | 0,828512 | 0,822133 | 0,830175 | 0,825587 |

Tabela 3.4: Podatkovna množica za metodo LastAction. V primerjavi z ostalimi podatkovnimi množicami vidimo, da je ta metoda boljša pri napovedovanju kratkoročnih interesov. Odebeljene so uspešnosti, ki so najvišje glede na istoležne celice v tabelah 3.1 - 3.4.

| zgodovina | starost profila | razred |
|---|---|---|
| 5 | 3 | c |
| 5 | 60 | d |
| 5 | 600 | b |
| 5 | 1800 | b |
| 5 | 3600 | b |
| . | . | . |
| . | . | . |
| . | . | . |
| 10 | 3 | c |
| 10 | 60 | d |
| 10 | 600 | d |
| . | . | . |
| . | . | . |
| . | . | . |
| 10 | 2419200 | a |
| . | . | . |
| . | . | . |
| . | . | . |
| 100 | 2419200 | a |

Tabela 3.5: Izsek preoblikovanih podatkov z namenom klasifikacije. V prvem stolpcu je dolžina hranjene zgodovine, v drugem starost profila in v tretjem koda metode, ki je pri tej kombinaciji parametrov najuspešnejša. Celotna množica ima 121 primerov.

# Poglavje 4

# Dinamična izbira metod

K iskanju najboljšega načina dinamične izbire metod smo pristopili s treh različnih smeri. Z uporabo klasifikacije, rangiranja z večznačnim učenjem in rangiranja s klasifikacijo. Pri uporabi rangiranja je bil namen podati urejen seznam predlogov. Zaradi slabših rezultatov smo rangiranje z večznačnim učenjem opustili.

## 4.1 Klasifikacija

Za namene klasifikacije smo uporabili java knjižnico za strojno učenje Weka [5]. Le-ta vsebuje veliko število implementacij različnih algoritmov za strojno učenje, kot so odločitvena drevesa, naivni Bayes, metode za meta-učenje itd. Poleg tega pa je na voljo tudi veliko razširitev, ki dodajajo nove algoritme ali izboljšujejo stare.

## 4.2 Uporabljeni učni modeli

Pri raziskovanju smo si ogledali naslednje algoritme:

- naivni Bayes,
- C4.5 odločitveno drevo,

- naključni gozd,
- logistična regresija,
- glasovanje, ki s povprečenjem verjetnosti kombinira algoritme J48, naivni Bayes in naključni gozd.

### 4.2.1 Naivni Bayes

Naivni Bayes [7] je preprost pristop h klasifikaciji, ki za svoje delovanje uporablja Bayesov teorem in sloni na dveh pomembnih predpostavkah. Predpostavlja namreč, da so vsi klasifikacijski atributi med seboj pogojno neodvisni in da neobstoječi podatki ne vplivajo na rezultat napovedi. Bayesova formula za računanje verjetnosti posameznega razreda:

$$p\left(c|a_1, a_2, ..., a_n\right) = p(c) * \prod_i \frac{p(c|a_i)}{p(c)} \tag{4.1}$$

Po formuli (4.1) naivni Bayes izračuna verjetnost razreda $p\left(c|a_1, a_2, ..., a_i\right)$ za kombinacijo parametrov $(a_1, a_2, ..., a_i)$. Za napoved izbere razred z najvišjo verjetnostjo.

### 4.2.2 Odločitveno drevo J48

J48 je java implementacija algoritma za grajenje odločitvenega drevesa C4.5 [10]. Odločitveno drevo je grajeno iz listov in vozlišč. Listi nam povedo razred, v katerega klasificiramo posamezen primer, v vozliščih pa izvajamo teste. Za vsak rezultat izvršenega testa obstaja veja oz. poddrevo. Klasificiramo tako, da začnemo v korenskem elementu in se premikamo skozi vozlišča, dokler nismo v listu. V vsakem vozlišču se na podlagi rezultata testa odločimo za poddrevo. Primer klasificiramo v razred, na katerega kaže list, v katerem smo pot zaključili. C4.5 za odločanje uporablja normaliziran informacijski dobitek.

### 4.2.3 Naključni gozd

Učenje z gozdovi je oblika učenja s kombiniranjem klasifikacijskih dreves. Vsa drevesa, ki so del gozda, nato glasujejo za najbolj popularni razred. Pri naključnem gozdu [2] so drevesa t.i. naključna drevesa. Naključno drevo zgradimo tako, da v vsakem vozlišču naključno izberemo manjšo množico atributov, po katerih bomo podatke delili v razrede. Gradimo po CART metodologiji do maksimalne velikosti drevesa in ne režemo (ang. pruning). Velikost množice atributov je konstantna.

### 4.2.4 Logistična regresija

Logistična regresija je statistični klasifikacijski model, ki za klasifikacijo uporablja logistično funkcijo (4.2). Na podlagi te funkcije se izračunajo verjetnosti posameznega razreda pri določeni kombinaciji atributov.

$$F(x_1, x_2, ..., x_k) = \frac{1}{1 + e^{-(ß_0 + ß_1 x_1 + ... + ß_k x_k)}} \tag{4.2}$$

### 4.2.5 Klasifikacija z večinskim razredom

Klasifikacija z večinskim razredom je najbolj preprost način klasifikacije. Algoritem prešteje vse pojavitve vseh razredov in nato vse nove primere klasificira v razred, ki vsebuje največ primerov. To metodo smo uporabili za postavitev referenčne vrednosti, ki jo morajo naši učni modeli dosegati. Če je ta mera presežena, lahko govorimo o uspešni dinamični izbiri metod.

Vrednosti $x_i$ so vrednosti atributov, $ß_i$ pa uteži. S prilagajanjem uteži lahko dosežemo dobro prilagajanje učni množici.

### 4.2.6 Kombiniranje z uporabo meta-učenja

Meta-učenje je metoda za kombiniranje več različnih enostavnih modelov učenja med seboj z namenom pridobitve novega, bolj uspešnega in fleksibilnega modela strojnega učenja. Enostavnejši modeli na učni množici podatkov

vrnejo napovedi za vsako kombinacijo atributov, te napovedi pa se uporabijo kot atributi pri meta-učenju. Algoritmi za meta-učenje uporabljeni in predstavljeni v tem diplomskem delu so stacking, bootstrap aggregating, boosting in voting.

### Metoda stacking

Metoda stacking [12] ali skladanje deluje tako, da združi vse napovedi uporabljenih učnih modelov s pomočjo združevalnega učnega modela. Vsi uporabljeni modeli najprej podajo napovedi za učno množico podatkov, ti podatki pa so uporabljeni pri učenju združevalnega modela. Zaradi slabše uspešnosti učenja rezultati metode stacking niso predstavljeni.

### Bootstrap aggregating

Algoritem bootstrap aggregating ali bagging [1] za učenje uporablja več enostavnejših učnih modelov. Učno množico podatkov razdeli v več naključno izbranih podmnožic, ki jih nato uporabi za gradnjo učnih modelov. Zgrajeni modeli za tem glasujejo za najbolj uspešno napoved. Primer učenja z uporabo algoritma bagging je tudi naključni gozd, opisan v 3.2.3.

### Boosting

Algoritem boosting [3] za meta učenje je v nekaterih primerih lahko uspešnejši od bagginga. Deluje tako, da zaporedoma gradi množico učnih modelov s poudarkom na napačno klasificiranih primerih. Poudarek dosežemo z uporabo uteži. Za vsakim zgrajenim modelom se podatki utežijo. Primeri, ki so napačno klasificirani pridobijo težo, med tem ko pravilno klasificirani primeri izgubijo težo. Tako dosežemo, da vsak naslednji klasifikator bolje klasificira do tedaj napačno klasificirane primere. Takšen način učenja lahko pomeni tudi, da se zgrajeni učni model preveč prilagaja podatkom. Zaradi slabše uspešnosti rezultati učenja z algoritmom boosting niso predstavljeni v rezultatih.

### Glasovanje

Glasovanje [9] je eden izmed preprostejših algoritmov za kombiniranje učnih modelov. Najprej zgradi več enostavnejših učnih modelov, ki nato med seboj glasujejo za najbolj uspešno napoved. Večinski razred se nato uporabi za klasifikacijo primera. Glasovanje se uporablja tudi pri algoritmu bagging opisanem v 3.4.2.

### 4.2.7 Ocenjevanje uspešnosti učnih modelov

Učne modele smo med seboj primerjali na podlagi več različnih parametrov. Najpomembnejša med njimi sta klasifikacijska točnost (CA) in povprečje uspešnosti profiliranja. Pri primerjanju uspešnosti klasifikacijskih rezultatov z rezultati, pridobljenimi z večznačnim učenjem, smo namesto CA uporabljali vrednosti priklic in preciznost. Vsi podatki pa so bili pridobljeni z uporabo 10-kratnega prečnega preverjanja.

#### Klasifikacijska točnost

CA je standardna mera za ocenjevanje uspešnosti klasifikacije, ki jo pridobimo z deljenjem števila pravilno klasificiranih primerov s številom vseh klasificiranih primerov in pomnožimo s 100.

$$CA = \frac{N_t}{N} * 100\% \tag{4.3}$$

#### Preciznost in priklic

Meri za preciznost in priklica smo uporabili za primerjanje rezultatov večznačnega učenja z rezultati klasifikacije.
Preciznost je število pravilno klasificiranih pozitivnih primerov (True Positive), deljeno s številom pravilno klasificiranih pozitivnih in napačno klasificiranih pozitivnih primerov (False Positive).

$$preciznost = \frac{T_p}{T_p + F_p} \tag{4.4}$$

Priklic je število pravilno klasificiranih pozitivnih primerov, deljeno s številom pravilno klasificiranih pozitivnih in napačno klasificiranih negativnih primerov.

$$priklic = \frac{T_p}{T_p + F_n} \tag{4.5}$$

### Povprečje uspešnosti profiliranja

Povprečja uspešnosti profiliranja smo pridobili s povprečenjem vrednosti uspešnosti profiliranja za napovedan razred preko vseh kombinacij parametrov zgodovina in starost profila. Uspešnosti profiliranja so vrednosti iz matrik podatkov, ki so nam bile na voljo za strojno učenje (Poglavje 3). Pridobljene so bile z uporabo posplošene mere kosinusne podobnosti [4].

S tem povprečjem smo pridobili dodatno mero uspešnosti učenja poleg klasifikacijske točnosti(CA), ki nam pomaga ugotoviti, kako uspešna je naša kombinirana profilirna metoda. Podatki niso enakomerno razporejeni, zato v nekaterih primerih prihaja do težave, kjer je CA visoka, a je bilo napačno klasificiranih nekaj ključnih primerov, ki znižujejo uspešnost profiliranja.

### Statistično testiranje

Dosežene izboljšave smo potrdili tudi s statističnim testom. Uporabili smo Wilcoxonov parni predznačeni test [11], ki je bil za testiranje uporabljen tudi v [6]. Testno statistiko izračunamo po formuli 4.6:

$$W = |\sum_{i=1}^{N_r}[sgn(x_{2,\mathrm{i}} - x_{1,\mathrm{i}}) * R_i]| \tag{4.6}$$

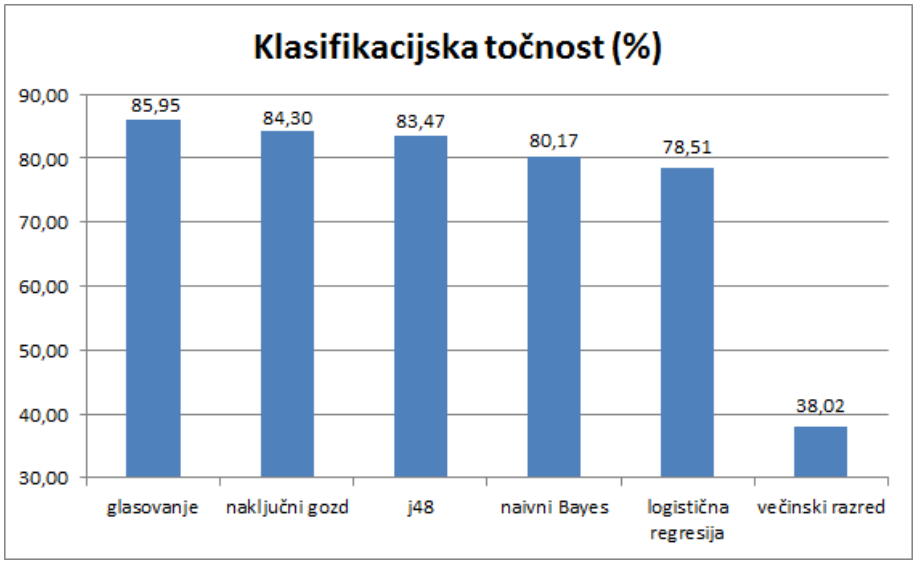
# Poglavje 5

# Rezultati

V nadaljevanju so predstavljeni rezultati najbolj uspešnih uporabljenih metod strojnega učenja. Vsi rezultati so pridobljeni z uporabo 10-kratnega prečnega preverjanja. Podatki so predstavljeni v obliki klasifikacijske točnosti in povprečja uspešnosti profiliranja. Prvi prikazujejo uspešnost samega strojnega učenja posameznega algoritma v primerjavi z drugimi, drugi pa uspešnost v primerjavi s statično uporabo metod profiliranja. Podana je tudi referenčna vrednost, ki pomeni klasifikacijo z uporabo večinskega razreda. Predstavljene so tudi primerjave z meritvami na drugi domeni podatkov in primerjave dinamične izbire s statičnimi vrednostmi vsake od metod profiliranja.

## 5.1 Klasifikacijska točnost

S slike 5.1 je razvidno, da je najuspešnejši pri učenju algoritem, ki z glasovanjem s povprečenjem verjetnosti kombinira algoritme J48, naivni Bayes in naključni gozd(85,9504 %). Sledi mu algoritem naključni gozd (84,2975 %). Nekaj slabše so se obnesli enostavnešji algoritmi J48 (83,4711 %), naivni Bayes (80,1653 %) in logistična regresija (78,5124 %), a kot bomo videli, so pri povprečjih uspešnosti profiliranja kljub temu precej uspešni. Klasifikacijska točnost z uporabo klasifikacije z večinskim razredom je 38,0165 % in je veliko nižja od vseh predstavljenih algoritmov.

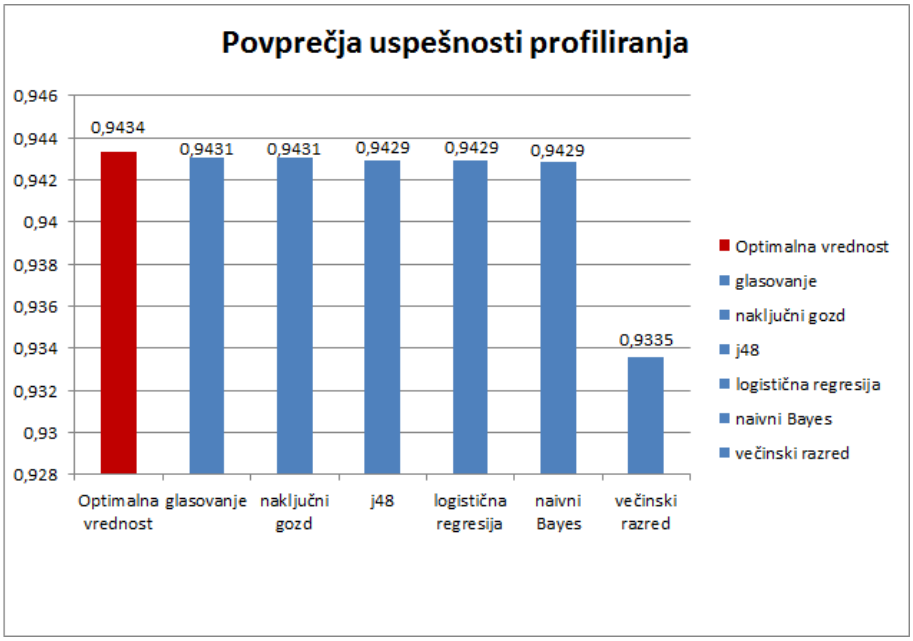



Slika 5.1: Klasifikacijske točnosti

## 5.2 Povprečja uspešnosti profiliranja

Na sliki 5.2 so prikazana povprečja uspešnosti profiliranja z uporabo dinamične izbire metod. **Podatke** smo pridobili tako, da smo pri vsaki napovedi za kombinacijo parametrov starost profila in zgodovina vzeli povprečno uspešnost iz tabel podatkov metod (tabele 3.1 - 3.4).
**Optimalno vrednost** smo izračunali s povprečenjem najbolj optimalnih napovedi izmed vseh štirih uporabljenih algoritmov pri vsaki kombinaciji parametrov starost profila in zgodovina.

Naš cilj je kar najbolj se približati optimalni vrednosti z dinamično izbiro metod profiliranja. Vidimo, da so razlike med posameznimi učnimi modeli veliko manjše kot pri uspešnosti učenja osnovnih modelov(slika 5.1). Še vedno je najboljši model glasovanje s povprečenjem verjetnosti modelov J48, naivni Bayes in naključni gozd, ki se od optimalne vrednosti (0,9434) razlikuje le za 0,0003. Prav tako se le za 0,0003 od optimalne vrednosti razlikuje algoritem naključni gozd. Za 0,0004 se od optimalne vrednosti razlikujeta napovedi




Slika 5.2: Povprečja uspešnosti profiliranja. Optimalna vrednost je bila pridobljena s povprečenjem najvišjih vrednosti v tabelah podatkov za vsako kombinacijo parametrov starost profila / zgodovina

algoritmov J48 in logistične regresije, sledi pa jima še napoved algoritma naivni Bayes, ki se od optimalne napovedi razlikuje za 0,0005. Referenčna vrednost, pridobljena z uporabo klasifikacije z večinskim razredom, se od optimalne napovedi razlikuje za 0,0098.

## 5.3 Izboljšave

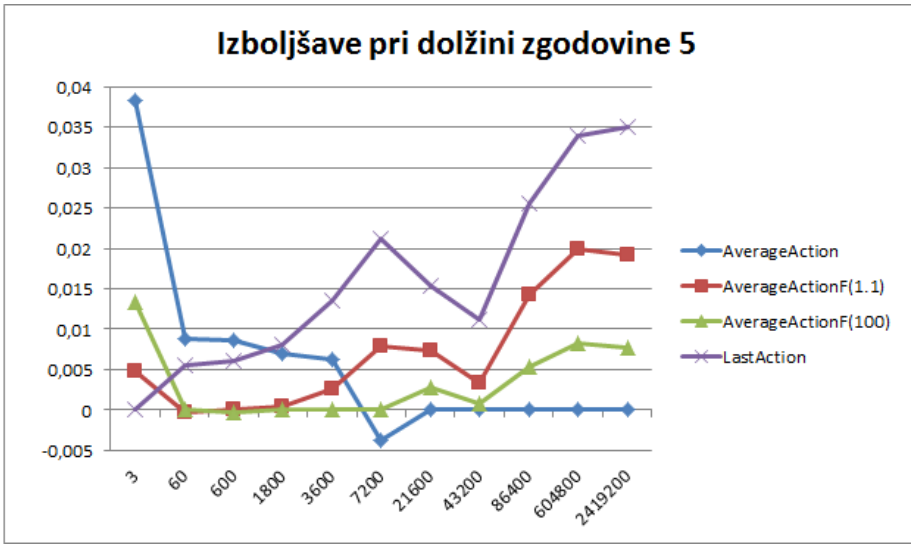
Na slikah 5.3 in 5.4 so prikazani grafi izboljšav, doseženih z dinamično izbiro metod z uporabo glasovanja s povprečenjem verjetnosti modelov J48, naivni Bayes in naključni gozd. **Graf izboljšav** prikazuje razliko v uspešnosti profiliranja med posamezno (statično) in našo dinamično metodo. Na osi x so starosti profilov v sekundah, na osi y pa razlike med statično in dinamično vrednostjo. Izrisani so rezultati za dolžino hranjene zgodovine uporabnika 5 in 100. Pri teh vrednostih so meje v uspešnostih profiliranja posameznih

algoritmov najbolj vidne. Na vsakem grafu lahko opazimo:

1. Visoko vrednost ordinate pri metodah, ki smo jih uspeli izboljšati s strojnim učenjem.

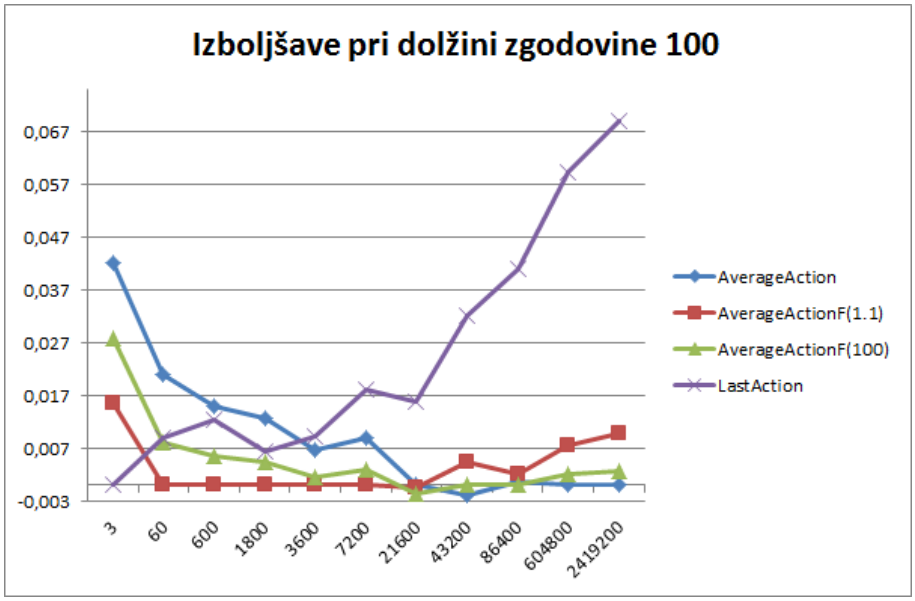
2. Vrednost ordinate 0 pri metodah, katerih uspešnost profiliranja je enaka dinamično izbrani metodi.

3. Negativno vrednost ordinate pri metodah, ki so bile napovedane napačno. Te vrednosti znižujejo uspešnost profiliranja.




Slika 5.3: Izboljšave pri dolžini zgodovine 5. Vidimo, da je metoda LastAction najbolj uspešna v prvih nekaj sekundah obiska, saj je na tej točki izboljšava zanjo enaka 0. Pri srednje dolgih obiskih je vidno kombiniranje metod AverageActionF(1.1) in AverageActionF(100). Kombiniranje je vidno pri izmenjavi dotikov osi x. Pri teh kombinacijah prihaja tudi do napačnih klasifikacij, ki pa zaradi majhnih razlik med metodama ne vplivajo usodno na končno uspešnost. Pri daljših obiskih je najuspešnejša metoda AverageAction.

Zelo dobro so razvidna območja, kjer so posamezne metode uspešne. Na slikah 5.3 in 5.4 lahko vidimo, da je metoda LastAction najbolj uspešna v prvih nekaj sekundah obiska, kmalu za tem pa ji uspešnost pade in jo tako z dinamično izbiro metod dopolnjujejo ostale. V srednjem območju, kjer je




Slika 5.4: Izboljšave pri dolžini zgodovine 100. Podobno kot pri grafu za zgodovino dolžine 5 je tudi tukaj v prvih sekundah najuspešnejša metoda LastAction. Pri srednje dolgih obiskih je v tem primeru vedno pravilno izbrana najuspešnejša metoda AverageActionF(1.1). Do napak pride pri obiskih dolžine 21600 in 43200 sekund. Napake zaradi majhne razlike med izbrano in najuspešnejšo metodo zopet niso usodne.

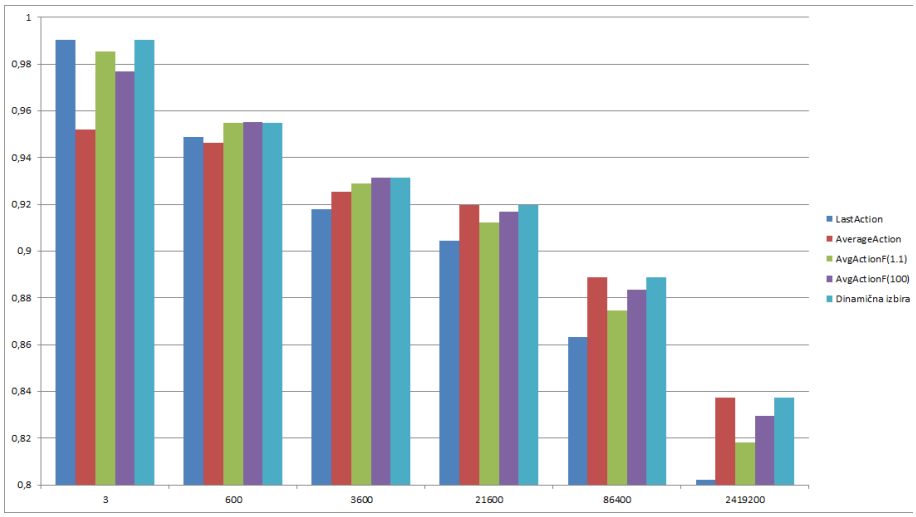
starost profila med 60 in 21600 sekundami, se prepletata metodi AverageActionF(1.1) in AverageActionF(100), kar na grafu vidimo po izmenjavanju velikosti izboljšav enakih 0(dotik osi x). Slednja je uspešna tudi pri daljših obiskih, kjer pa jo dopolnjuje metoda AverageAction, ki je najuspešnejša pri daljših obiskih.

Na sliki 5.5 so primerjane povprečne uspešnosti statičnih metod z novo dinamično metodo za zgodovino dolžine 5, na sliki 5.6 pa so prikazana odstopanja uspešnosti napovedanih vrednosti od optimalnih. Vidimo lahko, da so odstopanja minimalna v primerjavi z izboljšavami. Graf 5.6 prikazuje trend, da je napaka pri daljši zgodovini profila prisotna pri večji starosti profila. Ta trend znova upade pri zelo dolgih starostih profila. Do napak pride zaradi podobnosti v uspešnosti metod pri teh kombinacijah parametrov zgodovina / starost profila. Metode AverageAction, AverageActionF(100) in AverageAc-

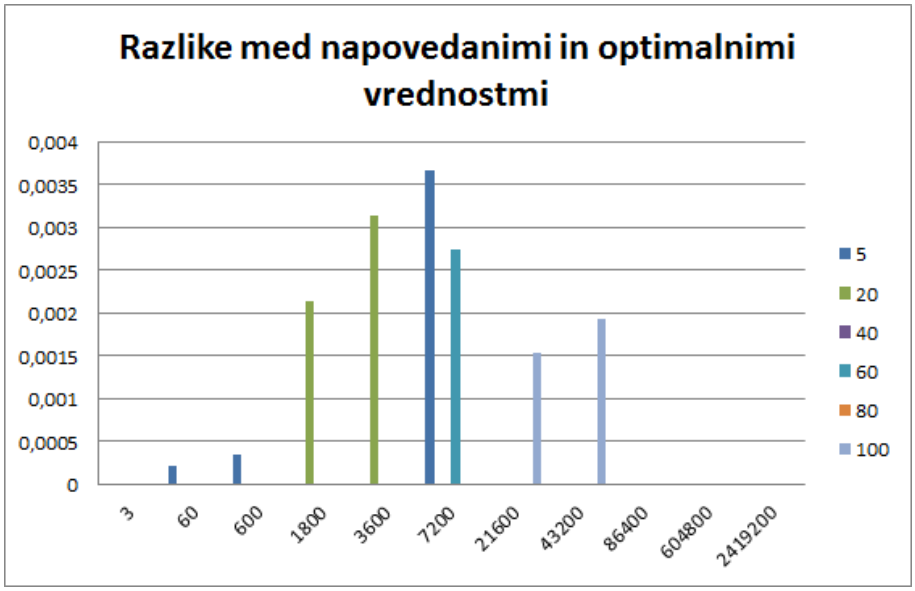
tionF(1.1) so, kot je razvidno tudi iz grafov izboljšav(5.3 in 5.4), v srednjem območju podatkov podobno uspešne.

Z uporabo dinamične izbire metod nam je uspelo izboljšati povprečno uspešnost profiliranja v primerjavi s posameznimi metodami.

- Metodo LastAction smo z izboljšali iz vrednosti 0,9184 na 0,9431 (razlika 0,0247, $p - vrednost < 2,2 * 10^{-16}$)
- Metodo AverageAction smo izboljšali iz vrednosti 0,9335 na 0,9431 (razlika 0,0096, $p - vrednost < 2,2 * 10^{-16}$)
- Metodo AverageActionF(100) smo izboljšali iz vrednosti 0,9384 na 0,9431 (razlika 0,0047, $p - vrednost < 2,2 * 10^{-16}$)
- Metodo AverageActionF(1.1) smo izboljšali iz vrednosti 0,9378 na 0,9430 (razlika 0,0052, $p - vrednost < 2,2 * 10^{-16}$)




Slika 5.5: Primerjava izboljšane dinamične metode s statičnimi metodami po uspešnosti pri dolžini zgodovine 5. Na osi x so starosti profilov, na y osi pa uspešnosti profilirnih metod. Vidimo, da ima naša dinamična metoda v večini primerov optimalno vrednost, napake pa so minimalne.




Slika 5.6: Razlike do optimalnih vrednosti. Na tem grafu vidimo odstopanja od optimalnih vrednosti. Na osi x so vrednosti razvrščene po starosti profila, na osi y pa po dolžini hranjene zgodovine. Razvidno je, da so v primerjavi z doseženimi izboljšavami odstopanja zelo majhna. Predvsem zaradi podobnosti med optimalno in dinamično izbrano vrednostjo kadar pride do napak. Graf prikazuje trend, da je napaka pri daljši zgodovini profila prisotna pri večji starosti profila. Ta trend znova upade pri zelo dolgih starostih profila.

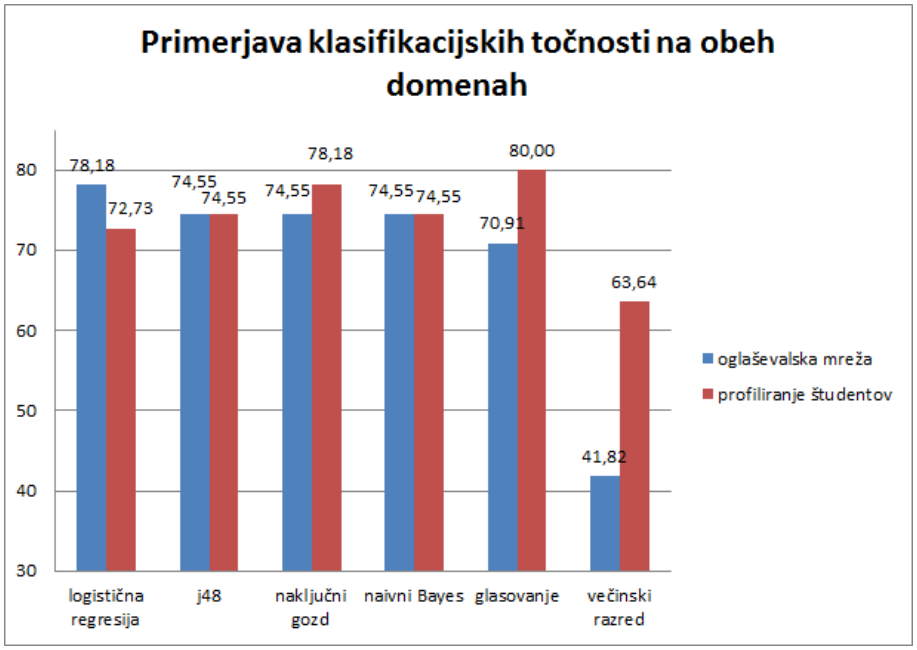
### 5.3.1 Statistični test

Rezultate smo preverili tudi z uporabo parnega Wilcoxonovega predznačnega testa. Izvedli smo 5 testov, po enega za vsako metodo in enega na združenih podatkih. Vse p-vrednosti so bile nižje od $2,2 * 10^{-16}$. Tako smo brez težav zavrnili ničelno hipotezo, da je razlika median na obeh vzorcih enaka 0 s stopnjo signifikantnosti $\alpha = 0,0001$. Test je bil izveden na množici parov velikosti $N = 6532888$, $N/4 = 1633222$ parov za vsako metodo.

## 5.4 Domena podatkov profiliranja študentov

Uspešnost učenja smo preverili tudi z uporabo izbranih učnih algoritmov na drugi domeni podatkov. Podatki za drugo domeno so bili pridobljeni s profiliranjem študentov uporabnikov spletne učilnice. Uspešnosti metod

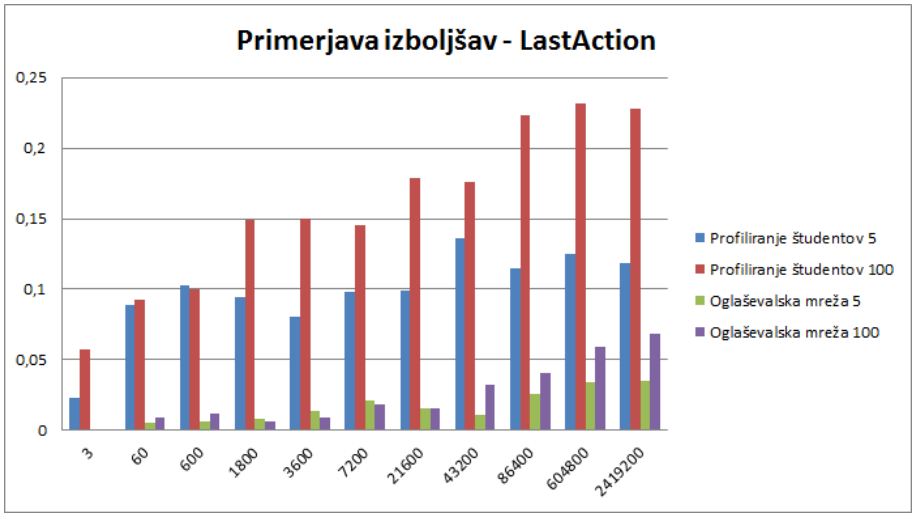
profiliranja so na tej domeni nižje in podatki bolj razpršeni, kar pomeni, da so tudi končne uspešnosti nižje in izboljšave dosežene z dinamično izbiro metod boljše. Uspešnosti učenja so primerljive s tistimi na prvi domeni podatkov (slika 5.7). Učenje za obe domeni je bilo izvedeno na zmanjšani množici podatkov, v kateri so bile vrednosti za dolžino zgodovine 5, 10, 20, 50 in 100.



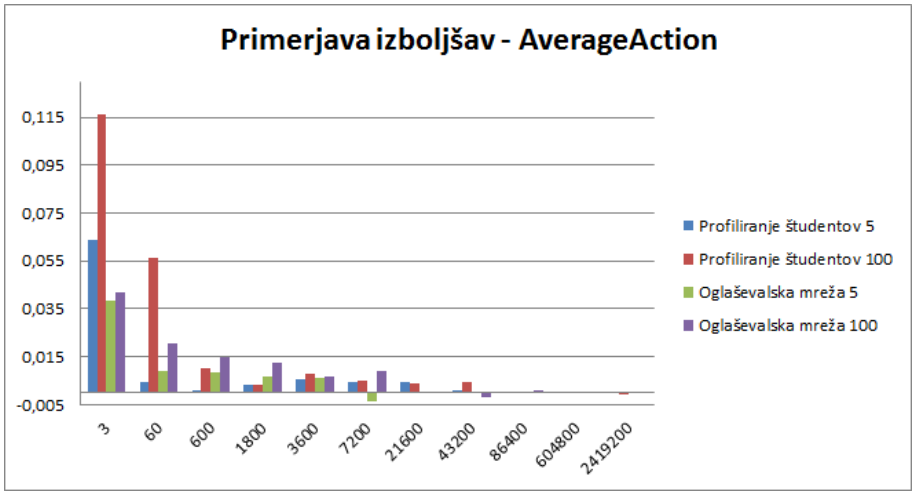

Slika 5.7: Primerjava uspešnosti učenja na obeh domenah podatkov. Na osi x so primerjani klasifikacijski algoritmi, na osi y pa vrednosti CA.

### 5.4.1 Primerjave izboljšav

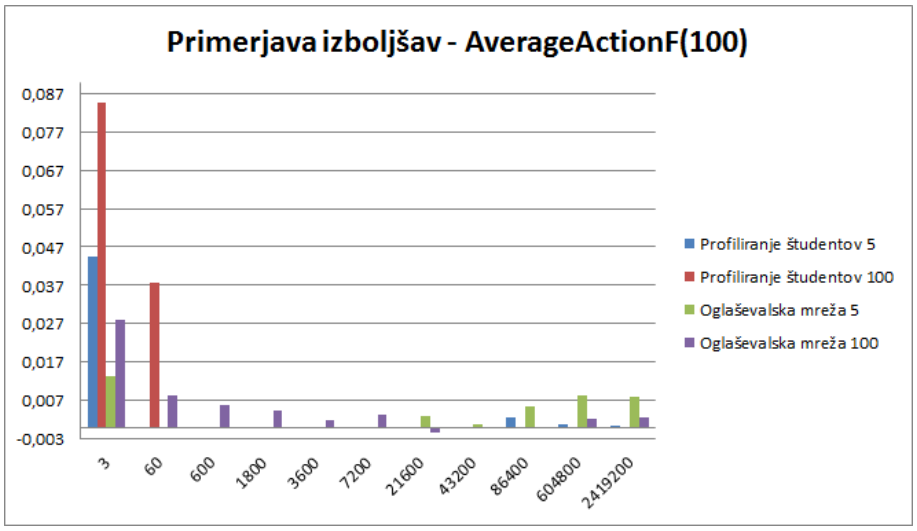
Na slikah z grafi 5.8, 5.9, 5.10 in 5.11 so združeni podatki o izboljšavah na obeh domenah za velikosti hranjene zgodovine uporabnika 5 in 100. Razvidno je, da so izboljšave metod z uporabo dinamične izbire na domeni profiliranja študentov precej večje kot pri domeni podatkov oglaševalske mreže. To je posledica predvsem večje razpršenosti podatkov na domeni profiliranja študentov ($\sigma_{\text{mdl}} = 0,0957, \sigma_{\text{adv}} = 0,0394$). To dodatno potrjuje uspešnost našega učenja in zgrajene dinamične metode profiliranja.



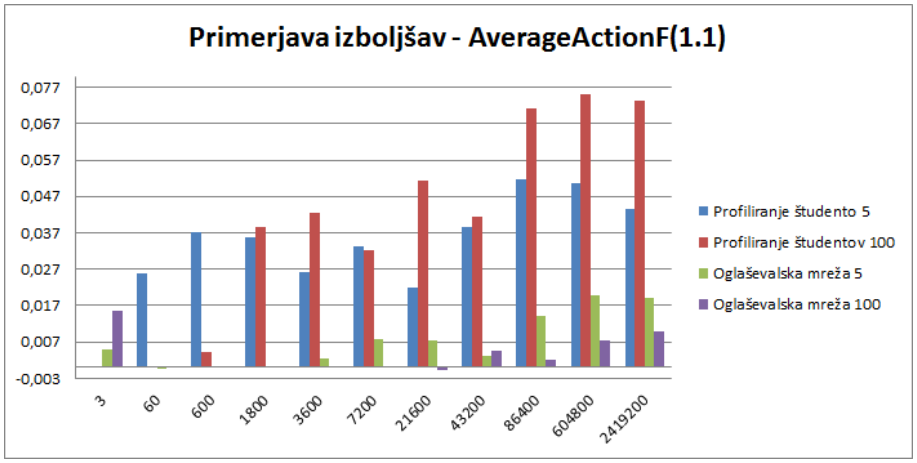

Slika 5.8: Primerjava izboljšav za metodo LastAction na obeh domenah. Vidimo, da se metoda LastAction obnaša podobno na obeh domenah podatkov. Najuspešnejša je v prvih nekaj sekundah obiska, nato pa so boljše druge metode.




Slika 5.9: Primerjava izboljšav za metodo AverageAction na obeh domenah. Na tem grafu vidimo, da se tudi Metoda AverageAction na obeh domenah obnaša podobno. Predvsem je uspešna pri dolgih obiskih.




Slika 5.10: Primerjava izboljšav za metodo AverageActionF(100) na obeh domenah. Razvidno je, da je metoda AverageActionF(100) najbolj uspešna pri srednje dolgih in dolgih obiskih. Izboljšave, ki jih dosežemo z uporabo dinamične izbire, so v teh primerih majhne.




Slika 5.11: Primerjava izboljšav za metodo AverageActionF(1.1) na obeh domenah. Metoda AverageActionF(1.1) je uspešna pri srednje dolgih obiskih predvsem na domeni oglaševalske mreže. V ostalih primerih so uspešnejše druge metode.

# Poglavje 6

# Zaključek

V tem diplomskem delu smo preučili in obdelali podatke uspešnosti profiliranja več profilirnih metod, opisanih v [6]. Glede na uspešnosti pri posamezni kombinaciji parametrov starost profila / zgodovina smo se odločili za štiri, ki so pri dinamični izbiri dajale najboljše rezultate. Te metode so LastAction, AverageAction, AverageActionF(100) in AverageActionF(1.1). Obdelane podatke smo nato uporabili pri strojnem učenju, s katerim smo dosegli dinamično izbiro metod. Preizkusili smo različne algoritme strojnega učenja. Za najuspešnejše so se izkazali klasifikacijski algoritmi J48, naivni Bayes in logistična regresija. Osnovne klasifikacijske algoritme smo poskušali izboljšati tudi z uporabo meta-učenja, a z omejeno uspešnostjo. Izboljšani klasifikatorji so dajali rahlo boljše rezultate, vendar so razlike premajhne, da bi lahko upravičili njihovo uporabo, saj so le-ti bolj kompleksni.

Z upoštevanjem preprostosti algoritmov in njihove uspešnosti, se je najbolje izkazal učni model J48. Poleg dobre dosežene klasifikacijske točnosti je tudi lahko razumljiv, preprost za izgradnjo in omogoča izdelavo if-then pravil, katera pa so zelo enostavna za implementacijo v kodi sami in ne zahtevajo dodatnih sredstev.

V nadaljnjem raziskovanju bi bilo smiselno bolj podrobno raziskati raz-

merja cen proti učinkovitosti algoritmov, saj zaključki glede kompleksnosti v tem delu temeljijo zgolj na grobih ocenah. Pri obdelavi podatkov je obetavne rezultate dajalo tudi filtriranje podatkov, kateremu se nismo podrobneje posvetili. Izboljšave so bile dosežene z odstranjevanjem podatkov za prvih nekaj sekund obiska. S tem preprečimo napačne napovedi, saj je očitno, da je v tem obdobju najuspešnejša metoda LastAction. Podobno bi bilo možno prenesti tudi na daljše obiske, kjer se ponavadi najbolj izkaže metoda AverageAction.

# Literatura


[1] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, August 1996.

[2] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[3] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996.

[4] Prasanna Ganesan, Hector Garcia-Molina, and Jennifer Widom. Exploiting hierarchical domain structure to compute similarity. *ACM Trans. Inf. Syst.*, 21(1):64–93, January 2003.

[5] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[6] Domen Košir. Web user profiles with time-decay and prototyping. *Applied Intelligence*, 2014 (v tisku).

[7] Igor Kononenko. Bayesian neural networks. *Biological Cybernetics*, 61(5):361–370.

[8] Igor Kononenko. *Strojno učenje*. Fakulteta za računalništvo in informatiko, 1997.

[9] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley and Sons, Inc., 2004.

[10] Steven L. Salzberg. C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3):235–240, 1994.

[11] Frank Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, December 1945.

[12] David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.